```
Assignment no. 04
Aim-
1. Linear Regression : Univariate and Multivariate
2. Least Square Method for Linear Regression
3. Measuring Performance of Linear Regression
4. Example of Linear Regression
5. Training data set and Testing data set
```

In [9]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [23]:
```
x=np.array([95,85,85,70,60])
y=np.array([85,90,70,64,70])
model= np.polyfit(x, y, 1)
model
```

Out[23]: array([0.53766234, 33.32467532])

In [24]:
```
predict = np.poly1d(model)
predict(65)
```

Out[24]: 68.27272727272727

In [25]:
```
y_pred= predict(x)
y_pred
```

Out[25]: array([84.4025974 , 79.02597403, 79.02597403, 70.96103896, 65.58441558])

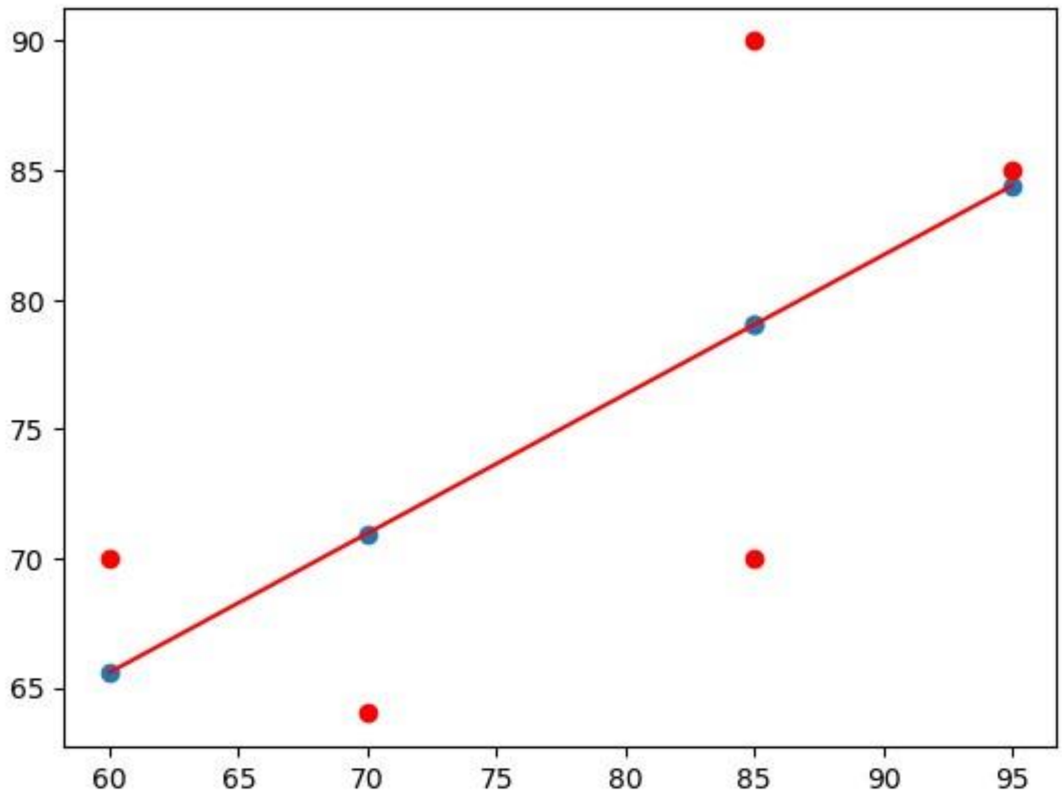
In [26]:
```
from sklearn.metrics import r2_score
r2_score(y, y_pred)
```

Out[26]: 0.4516887333445776

[27]:
```
y_line = model[1] + model[0]* x
plt.plot(x, y_line, c = 'r')
plt.scatter(x, y_pred)
plt.scatter(x,y,c='r')
```

Out[27]: <matplotlib.collections.PathCollection at 0x1e75c510c90>

In



[28]:
```
import ssl from sklearn.datasets import fetch_california_housing
ssl._create_default_https_context = ssl._create_unverified_context
california = fetch_california_housing(download_if_missing=True) X
= california.data y = california.target california
```

Out[28]:
```
{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,    2.55
555556,
          37.88      , -122.23      ],
       [   8.3014    ,   21.        ,    6.23813708, ...,    2.10984183,
 37.86      , -122.22      ],
       [   7.2574    ,   52.        ,    8.28813559, ...,    2.80225989,
          37.85      , -122.24      ],
       ...,
       [   1.7       ,   17.        ,    5.20554273, ...,    2.3256351 ,
 39.43      , -121.22      ],
       [   1.8672    ,   18.        ,    5.32951289, ...,    2.12320917,
 39.43      , -121.32      ],
       [   2.3886    ,   16.        ,    5.25471698, ...,    2.61698113,
          39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
  'AveRooms',
  'AveBedrms',
  'Population',
  'AveOccup',
  'Latitude',
  'Longitude'],
```

In

```
 'DESCR': '.. _california_housing_dataset:\n\nCalifornia Housing dataset\n
--------------------------\n\n**Data Set Characteristics:**\n\n    :Number
of Instances: 20640\n\n    :Number of Attributes: 8 numeric, predictive at
tributes and the target\n\n    :Attribute Information:\n        - MedInc
median income in block group\n        - HouseAge      median house age in
block group\n        - AveRooms      average number of rooms per household
\n        - AveBedrms     average number of bedrooms per household\n
- Population    block group population\n        - AveOccup      average nu
mber of household members\n        - Latitude      block group latitude\n
- Longitude     block group longitude\n\n    :Missing Attribute Values: No
ne\n\nThis dataset was obtained from the StatLib repository.\nhttps://www.
dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html\n\nThe target variable is
the median house value for California districts,\nexpressed in hundreds of
thousands of dollars ($100,000).\n\nThis dataset was derived from the 1990
U.S. census, using one row per census\nblock group. A block group is the s
mallest geographical unit for which the U.S.\nCensus Bureau publishes samp
le data (a block group typically has a population\nof 600 to 3,000 peopl
e).\n\nA household is a group of people residing within a home. Since the
average\nnumber of rooms and bedrooms in this dataset are provided per hou
sehold, these\ncolumns may take surprisingly large values for block groups
with few households\nand many empty houses, such as vacation resorts.\n\nI
t can be downloaded/loaded using the\n:func:`sklearn.datasets.fetch_califo
rnia_housing` function.\n\n.. topic:: References\n\n    - Pace, R. Kelley
and Ronald Barry, Sparse Spatial Autoregressions,\n      Statistics and Pr
obability Letters, 33 (1997) 291-297\n'}
```

[29]:

```
data = pd.DataFrame(california.data)
data.columns = california.feature_names
data.head()
```

Out[29]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25

In [30]:

```
data['PRICE'] = california.target
data.isnull().sum()
```

Out[30]:

```
MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
PRICE         0
dtype: int64
```

In

In [31]:
```
data.isnull().sum() from sklearn.model_selection
import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(X, y, test_size=0.2, random_
import sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression() model=lm.fit(xtrain,
ytrain)
```

In [32]:
```
ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
df=pd.DataFrame(ytrain_pred,ytrain)
df=pd.DataFrame(ytest_pred,ytest)
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(ytest, ytest_pred) print(mse)
mse = mean_squared_error(ytrain_pred,ytrain)
print(mse)
```

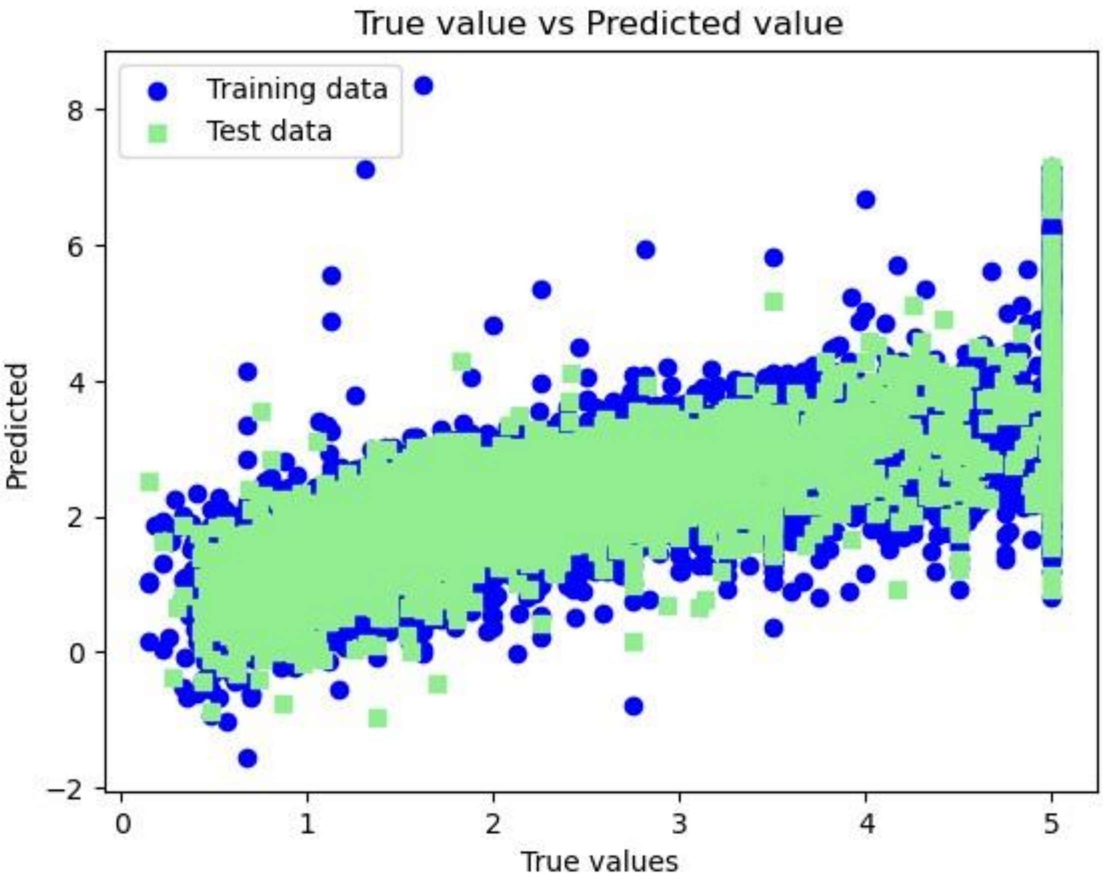
```
0.5289841670367244
0.5234413607125447
```

In [33]:
```
mse = mean_squared_error(ytest, ytest_pred)
print(mse)
```

```
0.5289841670367244
```

In [34]:

```
plt.scatter(ytrain ,ytrain_pred,c='blue',marker='o',label='Training data')
plt.scatter(ytest,ytest_pred ,c='lightgreen',marker='s',label='Test data')
plt.xlabel('True values')
plt.ylabel('Predicted')
plt.title("True value vs Predicted value")
plt.legend(loc= 'upper left')
#plt.hlines(y=0,xmin=0,xmax=50)
plt.plot()
plt.show()
```




```
Name- Sharayu Nanaware  Rollno:- 13243
```